

Design and Analysis of Algorithms Assignment 5

Harrison Lee, Alex Zhao

March 15, 2019

Problem 7.6. *Lightswitch Mapping*

We are given n light fixtures and n light switches in a house. We are also given a subroutine that determines in $O(1)$ time whether two lines intersect.

We can use this subroutine to determine in $O(1)$ time if a wall is blocking a fixture and switch pair by inputting the wall as one line segment and the fixture and switches' coordinates as the end points of the other line segment input.

Give an algorithm that determines if a given floor plan is ergonomic, or can be mapped so all light fixtures are visible with all light switches, in polynomial n and m time.

Algorithm 1. *Determine in $O(mn)$ time if the given floor plan can be ergonomic.*

Proof. First, make a bipartite graph of all n switches to all n fixtures. Define edges as all pairs of switches and fixtures that are visible, with capacities of 1. Pairs that are not visible to each other effectively have 0 capacity or no edge.

Next, create source and sink nodes, connecting them to all switch and fixture nodes respectively with edges of capacity 1, to simulate supplies and demands of 1 light fixture control connection.

This graph is ready for Ford-Fulkerson to be run on it, as discussed in lecture.

Running Time: Making initial bipartite graph: n^2 for making each node and checking its visibility with each other node, times $m * 1$ for checking against each other wall using the given $O(1)$ function. Initial total runtime of $O(n^2m)$.

Creating the source and sink node links can be done at the same time the n switches and fixtures are initialized, so they shouldn't have a significant impact on run time.

Ford-Fulkerson runs in $O(\#edges * max_flow)$. Max flow is 1, as all edges are given that value. There are at most $n * (n + 2)$ edges, as each fixture-switch pair can share an edge, in addition to the edges connecting the source and sink nodes. These result in an added $O(n^2)$.

Total runtime = $O(n^2m + n^2) = O(n^2m)$.

□

Problem 7.20. Balloon Matching

- a. Apply generalized matching as described in class with balloons on the source side and conditions on the sink side. An edge from a balloon to a condition indicates that that balloon can measure that condition.

Let the capacity of the edges from the super source to the balloons be 2, the capacity of the edges from the balloons to the conditions be 1, and the lower bound on the edges from the conditions to the super sink be k .

Solve by reducing to max-flow and using Ford-Fulkerson as done in class. There is a valid solution if there exists a flow of size $m \cdot k$. The runtime of Ford-Fulkerson is $O(\text{nodes} \cdot \text{edges})$

- b. Create three additional nodes for each condition node, one for each subcontractor. If a balloon can measure a condition, add an edge from that balloon to the subcontractor node of that condition (still capacity 1). Then from each condition's subcontractor nodes, create an edge of capacity $k-1$ to the condition node. Edges from the super source to balloon nodes are still capacity 2 and edges from condition nodes to the super sink are still lower bound k .

Now each condition node only receives flow from its subcontractor nodes but each subcontractor node can only provide $k-1$ flow, meaning at least two subcontractor nodes need to contribute to the flow to a condition node for it to be satisfied.

Then reduce and run Ford-Fulkerson as before.